UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/683,929	10/09/2003	John W. Rapp	1934-13-3	2222

7590 11/13/2008

Bryan A. Santarelli
GRAYBEAL JACKSON HALEY LLP
Suite 350
155 - 108th Avenue NE
Bellevue, WA 98004-5901

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	PAPER NUMBER
2183	
MAIL DATE	DELIVERY MODE
11/13/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/683,929	RAPP ET AL.	
	Examiner	Art Unit	
	DAVID J. HUISMAN	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *27 October 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-16,41-50 and 66-85* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☒ Claim(s) *74 and 81* is/are allowed.

6)☒ Claim(s) *1-13,15,16,41-47,49,50,68-72,75-79 and 82-85* is/are rejected.

7)☒ Claim(s) *14,48,66,67,73 and 80* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *11 March 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-16, 41-50, and 66-85 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: After-Final Amendment as received on 10/27/2008.

Specification

3. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed. The examiner recommends --Pipeline Accelerator that Receives or Transmits Data Via a Message and Processes the Data Without Executing a Program Instruction--.

Claim Objections

4. Please address the following minor informalities:

- In claim 6, line 4, replace "comprising," with --comprising:--.
- In claim 8, line 5, replace "to," with --to:--.
- In claim 74, on page 15, line 1, replace "to," with --to:--.
- In claim 75, on page 16, line 1, replace "to," with --to:--
- In claim 78, on page 17, last line, insert a colon after "to".
- In claim 79, on page 18, 2nd to last line, insert --the-- before "destination".
- In claim 79, on page 19, line 1, insert a colon after "to".

Appropriate correction is required.

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-5, 7, 9-10, 12, 15-16, 41-45, 49-50, 69-72, 76-79, and 83-85 are rejected under 35 U.S.C. 103(a) as being unpatentable over Krishna et al., U.S. Patent Application Publication No. US 2003/0014627 A1 (herein referred to as Krishna), in view of Yin, U.S. Patent No. 6,028,939. Furthermore, Wu et al., "CryptoManiac: A Fast Flexible Architecture for Secure Communication", 2001, has been cited as extrinsic evidence showing that fast hardware implementations of cryptography functions are known in the art.

7. Referring to claim 1, Krishna has taught a pipeline accelerator, comprising:

a) a memory. See Fig.3, at least components 312, 302, and 318, all of which may be collectively referred to as "a memory".

b) a hardwired-pipeline circuit coupled to the memory, including at least one processing pipeline (see Fig.2 and Fig.3, and at least paragraphs [0023], [0044], and [0097]), and operable to:

b1) receive a message that includes data and that includes a header having information indicating a destination of the data by receiving the data and the information on at least one common bus line. See Figs.2-3 and note that a message is received by the input

FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed.

b2) extract the data from the message, load the extracted data into the memory, retrieve the extracted data from the memory, and process the retrieved data with a pipeline corresponding to the destination. See paragraphs [0038] and [0049] an note that the message is parsed such that the data is extracted from the message and stored in buffers 312 (Fig.3). The data is then retrieved from said buffers and processed by crypto-engines 316 (Fig.3).

b3) provide the processed data to an external source. See Fig.3, and note that after processing, the processed data is sent to an external source by way of output FIFO.

b4) Krishna has not explicitly taught that the retrieved data is processed without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more

limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

8. Referring to claim 2, Krishna in view of Yin has taught that pipeline accelerator of claim 1. Krishna has not explicitly taught that the memory is disposed on a first integrated circuit and the pipeline circuit is disposed on a second integrated circuit. However, the separation of components into distinct integrated circuits is not a patentable feature. One of ordinary skill in the art would have recognized that separation of components onto separate chips allows for increased flexibility and reduced repair cost because if just a portion of a system is to be upgraded or repaired, then only that integrated circuit would be upgraded or repaired, as opposed to upgrading or repairing the whole system (if it were on a single chip). As a result, it would have been obvious to modify Krishna such that the memory of the accelerator of Fig.2-3 is implemented on a separate integrated circuit from that of the pipeline processing circuit. Also, see *Nerwin v. Erlichman 168 USPQ 177 (1969).*

9. Referring to claim 3, Krishna in view of Yin has taught the pipeline accelerator of claim 1, wherein the pipeline circuit is disposed on a field-programmable gate array. See Yin, column 11, lines 1-9.

10. Referring to claim 4, Krishna in view of Yin has taught the pipeline accelerator of claim 1 wherein the pipeline circuit is operable to provide the processed data to the external source by: loading the processed data into the memory; retrieving the processed data from the memory; and providing the retrieved processed data to the external source. See Fig.3 and note that after processing, the processed data is loaded into and retrieved from output memory 318 and then provided to the external source.

11. Referring to claim 5, Krishna in view of Yin has taught the pipeline accelerator of claim 1, wherein the external source comprises a processor; and the pipeline circuit is operable to receive the data from the processor. See paragraphs [0004], [0010], and [0030]. The purpose of the accelerator is to receive a message from a first processor on a network, encrypt or decrypt the data, and then send it to another processor over the network.

12. Referring to claim 7, Krishna has taught a pipeline accelerator, comprising:

a) a memory. See Fig.3, at least components 312, 302, and 318, all of which may be collectively referred to as "a memory".

b) a hardwired-pipeline circuit coupled to the memory (see Fig.2 and Fig.3, and at least paragraphs [0023], [0044], and [0097]), and operable to:

b1) receive data. See Figs.2-3 and note that a data packet is received.

b2) process the received data, load the processed data into the memory, and retrieve the processed data from the memory. See Fig.3 and note that the data is processed by crypto-engines 316 before being stored in and ultimately retrieved from output memory 318.

b3) Krishna has not explicitly taught that the retrieved data is processed without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such

that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

b4) Krishna has further taught providing the processed data to an external source (see Fig.3), but has not explicitly taught generating a message header that includes first information indicating a destination of the processed data, generating a message that includes the processed data and the header, and then providing the message to the external source. However, this is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node.

13. Referring to claim 9, Krishna has taught a pipeline accelerator comprising:

a) first and second memories. See 3, and note first memory 312 and second memory 318.

b) a hardwired-pipeline circuit (see Fig.3, at least components 316) coupled to the first and second memories and comprising:

b1) an input-data handler operable to receive from an external source a first message that includes raw data and that includes a first header having information indicating a destination of the raw data, to extract the raw data from the message, and to load the raw data into the first memory. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed. The

message is parsed such that the data is extracted from the message and stored in first memory 312 (Fig.3).

b2) at least one hardwired pipeline operable to process data. See at least the crypto-engines 316 of Fig.3. Note from at least paragraphs [0023], [0044], and [0097], that the system is pipelined.

b3) a pipeline interface operable to retrieve the raw data from the first memory, provide the retrieved raw data to the hardwired pipeline corresponding to the destination, and load processed data from the hardwired pipeline into the second memory. See Fig.3, and note that before the data is processed by units 316, it must be retrieved from the first memory 312. The portion of the system retrieving the data would be the pipeline interface. After processing, the data is loaded into the second memory 318.

b4) an output-data handler operable to retrieve the processed data from the second memory, to generate a second header having first information indicating a destination of the processed data, to generate a second message that includes the processed data and the second header, and to provide the second message to the external source via at least one same bus line. This is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node.

b5) Krishna has not explicitly taught that the at least one hardwired pipeline is operable to process data without executing a program instruction. However, it should be noted

that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

14. Referring to claim 10, Krishna in view of Yin has taught a pipeline accelerator as described in claim 9 wherein the first and second memories each include respective first and second ports, the input-data handler is operable to load the raw data via the first port of the first memory, the pipeline interface is operable to retrieve the raw data via the second port of the first memory and to load the processed data via the first port of the second memory, and the output-data handler is operable to retrieve the processed data via the second port of the second memory. See Fig.3.

15. Referring to claim 12, Krishna in view of Yin has taught that pipeline accelerator of claim 9 wherein the pipeline circuit is disposed on a field-programmable gate array (see Yin, column 11, lines 1-9). Krishna has not explicitly taught that the first and second memories are respectively disposed on first and second integrated circuits. However, the separation of components into distinct integrated circuits is not a patentable feature. One of ordinary skill in the art would have recognized that separation of components onto separate chips allows for increased flexibility and reduced repair cost because if just a portion of a system is to be upgraded or repaired, then only that integrated circuit would be upgraded or repaired, as opposed to upgrading or repairing the whole system (if it were on a single chip). As a result, it would have been obvious to modify Krishna such that that the first and second memories are respectively disposed on first and second integrated circuits. Also, see *Nerwin v. Erlichman 168 USPQ 177 (1969).*

16. Referring to claim 15, Krishna in view of Yin has taught a pipeline accelerator as described in claim 9.

a) Krishna in view of Yin has further taught that each of the input-data handler, hardwired pipeline, pipeline interface, and output-data handler has a respective operating configuration. That is, each component in Krishna operates in a specific fashion and consequently, each component inherently has a respective operating configuration.

b) Krishna in view of Yin has further taught a configuration manager coupled to and operable to set the operating configurations of the input-data handler, hardwired pipeline, pipeline interface, and output-data handler. That is, it is the inherent nature of an FPGA to be coupled to a configuration manager so that the FPGA may be programmed to include the desired functionality.

17. Referring to claim 16, Krishna in view of Yin has taught a pipeline accelerator as described in claim 9.

a) Krishna in view of Yin has inherently taught that each of the input-data handler, hardwired pipeline, pipeline interface, and output-data handler has a respective operating status. That is, at the very least, each component in the system is either operating or not operating (and these are statuses).

b) Krishna in view of Yin has not taught an exception manager coupled to and operable to identify an exception in the input-data handler, hardwired pipeline, pipeline interface, or output-data handler in response to the operating statuses. However, Official Notice is taken that checking for errors in a pipeline during processing is well known and accepted in the art. Any type of error which causes an exception should be monitored so that the system can take appropriate action to correct the error. As a result, in order to ensure proper execution, it would

have been obvious to one of ordinary skill in the art at the time of the invention to modify Krishna in view of Yin such that an exception manager is implemented to detect exceptions.

18. Referring to claim 41, the method of claim 41 is performed by the circuit of claim 1. Consequently, claim 41 is rejected for the same reasons set forth in the rejection of claim 1. Also, the message has information indicating a size of the message. See page 10 and paragraphs [0094]-[0095] and note that the data to be processed includes a byteCount, which indicates the size of the payload to be processed.

19. Referring to claim 42, Krishna in view of Yin has taught a method as described in claim 41. Furthermore, the method of claim 42 is performed by the circuit of claim 4. Consequently, claim 42 is rejected for the same reasons set forth in the rejection of claim 4.

20. Referring to claim 43, the method of claim 43 is performed by the accelerator of claim 7. Consequently, claim 43 is rejected for the same reasons set forth in the rejection of claim 7. In addition, the providing the message to an external source comprises providing on a single bus. See Fig.3 and note that the processed data is outputted via a single bus.

21. Referring to claim 44, the method of claim 44 is performed by the circuit of claim 9. Consequently, claim 44 is rejected for the same reasons set forth in the rejection of claim 9.

22. Referring to claim 45, Krishna in view of Yin has taught a method as described in claim 44. Furthermore, the method of claim 45 is performed by the circuit of claim 10. Consequently, claim 45 is rejected for the same reasons set forth in the rejection of claim 10.

23. Referring to claim 49, Krishna in view of Yin has taught a method as described in claim 44. Furthermore, Krishna has taught setting parameters for loading and retrieving the raw data, processing the retrieved data, and loading and providing the processed data. See paragraph

[0038], for instance, and note that encryption keys must be set (DES inherently uses a key for encryption). Also, pointers need to be set to load and store data from/to buffers/queues, etc.

24. Referring to claim 50, Krishna in view of Yin has taught a method as described in claim 44. While Krishna has not explicitly taught determining whether an error occurs during the loading and retrieving of the raw data, the processing of the retrieved data, and the loading and providing of the processed data, Official Notice is taken that checking for errors during processing is well known and accepted in the art. Any type of error, which causes an exception, should be monitored so that the system can take appropriate action to correct the error. As a result, in order to ensure proper execution, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Krishna to determining whether an error occurs during the loading and retrieving of the raw data, the processing of the retrieved data, and the loading and providing of the processed data.

25. Referring to claim 69, Krishna in view of Yin has taught the pipeline accelerator as described in claim 7, wherein the hardwired-pipeline circuit is further operable to:

a) store in association with the processed data second information indicating the destination of the processed data. This is deemed inherent as when a first processor sends data over a network to a second processor, the destination is included in the header. Therefore, when this destination is received by the accelerator, it must be stored while the encryption occurs so that when the data is encrypted, the data is again sent on towards its destination with the destination information.

b) generate the message header in response to the second information. Again, this is inherent for the reasons stated above. Essentially, after the encryption occurs, the data will be sent on

towards its destination. Hence, as is known, a message header including the destination will be attached to the data.

26. Referring to claim 70, Krishna in view of Yin has taught a pipeline accelerator as described in claim 69 wherein the second information equals the first information. From the source until the final destination, the destination information in all headers will be the same to ensure that the data goes to the correct destination. Hence, even if data needs to be encrypted on the way, the destination will stay the same.

27. Referring to claim 71, Krishna has taught a pipeline accelerator, comprising:

a) a memory. See Fig.3, at least components 312, 302, and 318, all of which may be collectively referred to as "a memory".

b) a hardwired-pipeline circuit coupled to the memory (see Fig.2 and Fig.3, and at least paragraphs [0023], [0044], and [0097]), and operable to:

b1) receive data. See Figs.2-3 and note that a data packet is received.

b2) process the received data, load the processed data into the memory, and retrieve the processed data from the memory. See Fig.3 and note that the data is processed by crypto-engines 316 before being stored in and ultimately retrieved from output memory 318.

b3) Krishna has not explicitly taught that the retrieved data is processed without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well

known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

b4) Krishna has further taught providing the processed data to an external source (see Fig.3), but has not explicitly taught generating a message header that includes first information indicating a destination of the processed data, generating a message that includes the processed data and the header, and then providing the message to the external source. However, this is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is

located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node.

b5) Krishna has also not explicitly taught storing a pointer to the processed data, storing in association with the pointer second information indicating the destination of the processed data, retrieving the processed data in response to the pointer, and generating the message header in response to the second information. However, this is deemed inherent in Krishna given the output queue of Fig.3. Specifically, data in a queue is accessed via a head pointer. hence, the head pointer must be stored and used to retrieve the data prior to sending it out. Also, the ultimate destination of the data must be stored, otherwise the accelerator will not know where to send the data to.

28. Referring to claim 72, Krishna has taught a pipeline accelerator, comprising:

a) a memory. See Fig.3, at least components 312, 302, and 318, all of which may be collectively referred to as "a memory".

b) a hardwired-pipeline circuit coupled to the memory (see Fig.2 and Fig.3, and at least paragraphs [0023], [0044], and [0097]), and operable to:

b1) receive data. See Figs.2-3 and note that a data packet is received.

b2) process the received data, load the processed data into the memory, and retrieve the processed data from the memory. See Fig.3 and note that the data is processed by crypto-engines 316 before being stored in and ultimately retrieved from output memory 318.

b3) Krishna has not explicitly taught that the retrieved data is processed without executing a program instruction. However, it should be noted that the purpose of

Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

b4) Krishna has further taught providing the processed data to an external source (see Fig.3), but has not explicitly taught generating a message header that includes first

information indicating a destination of the processed data, generating a message that includes the processed data and the header, and then providing the message to the external source. However, this is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node.

b5) Krishna has also not explicitly taught storing a pointer to the processed data in a location associated with the destination of the processed data, retrieving the processed data in response to the pointer, and generating the message header in response to the second information. However, this is deemed inherent in Krishna given the output queue of Fig.3. Specifically, data in a queue is accessed via a head pointer. hence, the head pointer must be stored and used to retrieve the data prior to sending it out. Also, the ultimate destination of the data must be stored, otherwise the accelerator will not know where to send the data to.

29. Referring to claim 76, Krishna has taught a pipeline accelerator, comprising:

a) first and second memories. See 3, and note first memory 312 and second memory 318.

b) a hardwired-pipeline circuit (see Fig.3, at least components 316) coupled to the first and second memories and comprising:

b1) an input-data handler operable to receive from an external source a first message that includes raw data and that includes a first header having information indicating a destination of the raw data, to extract the raw data from the message, and to load the raw

data into the first memory. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed. The message is parsed such that the data is extracted from the message and stored in first memory 312 (Fig.3).

b2) at least one hardwired pipeline operable to process data. See at least the crypto-engines 316 of Fig.3. Note from at least paragraphs [0023], [0044], and [0097], that the system is pipelined.

b3) a pipeline interface operable to retrieve the raw data from the first memory, provide the retrieved raw data to the hardwired pipeline corresponding to the destination, and load processed data from the hardwired pipeline into the second memory. See Fig.3, and note that before the data is processed by units 316, it must be retrieved from the first memory 312. The portion of the system retrieving the data would be the pipeline interface. After processing, the data is loaded into the second memory 318.

b4) an output-data handler operable to retrieve the processed data from the second memory, to generate a second header having first information indicating a destination of the processed data, to generate a second message that includes the processed data and the second header, and to provide the second message to the external source via at least one same bus line. This is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a

network, a message header with destination information must be generated such that the data is sent to the appropriate node.

b5) wherein the pipeline interface is further operable to store in association with the processed data second information indicating the destination of the processed data. This is deemed inherent as when a first processor sends data over a network to a second processor, the destination is included in the header. Therefore, when this destination is received by the accelerator, it must be stored while the encryption occurs so that when the data is encrypted, the data is again sent on towards its destination with the destination information.

b6) wherein the output-data handler is further operable to generate the first information from the second information. Again, this is inherent for the reasons stated above. Essentially, after the encryption occurs, the data will be sent on towards its destination. Hence, as is known, a message header including the destination (second information, which is the same as the first information) will be attached to the data.

b7) Krishna has not explicitly taught that the at least one hardwired pipeline is operable to process data without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES

processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

30. Referring to claim 77, Krishna in view of Yin has taught the pipeline accelerator of claim 76 wherein the second information equals the first information. From the source until the final destination, the destination information in all headers will be the same to ensure that the data goes to the correct destination. Hence, even if data needs to be encrypted on the way, the destination will stay the same.

31. Referring to claim 78, Krishna has taught a pipeline accelerator, comprising:

a) first and second memories. See 3, and note first memory 312 and second memory 318.

b) a hardwired-pipeline circuit (see Fig.3, at least components 316) coupled to the first and second memories and comprising:

b1) an input-data handler operable to receive from an external source a first message that includes raw data and that includes a first header having information indicating a destination of the raw data, to extract the raw data from the message, and to load the raw data into the first memory. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed. The message is parsed such that the data is extracted from the message and stored in first memory 312 (Fig.3).

b2) at least one hardwired pipeline operable to process data. See at least the crypto-engines 316 of Fig.3. Note from at least paragraphs [0023], [0044], and [0097], that the system is pipelined.

b3) a pipeline interface operable to retrieve the raw data from the first memory, provide the retrieved raw data to the hardwired pipeline corresponding to the destination, and load processed data from the hardwired pipeline into the second memory. See Fig.3, and note that before the data is processed by units 316, it must be retrieved from the first memory 312. The portion of the system retrieving the data would be the pipeline interface. After processing, the data is loaded into the second memory 318.

b4) an output-data handler operable to retrieve the processed data from the second memory, to generate a second header having first information indicating a destination of the processed data, to generate a second message that includes the processed data and the

second header, and to provide the second message to the external source via at least one same bus line. This is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node.

b5) Krishna has not explicitly taught that the pipeline interface is further operable to store a pointer to the processed data, store in association with the pointer second information indicating the destination of the processed data, wherein the output-data handler is further operable to retrieve the processed data in response to the pointer, and generating the first information from the second information. However, this is deemed inherent in Krishna given the output queue of Fig.3. Specifically, data in a queue is accessed via a head pointer. hence, the head pointer must be stored and used to retrieve the data prior to sending it out. Also, the ultimate destination of the data must be stored, otherwise the accelerator will not know where to send the data to. Note also that the first information (outbound destination) must be the same as the second destination (incoming destination) so that the packet ultimately reaches the desired destination.

b6) Krishna has not explicitly taught that the at least one hardwired pipeline is operable to process data without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim

20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

32. Referring to claim 79, Krishna has taught a pipeline accelerator, comprising:

a) first and second memories. See 3, and note first memory 312 and second memory 318.

b) a hardwired-pipeline circuit (see Fig.3, at least components 316) coupled to the first and second memories and comprising:

b1) an input-data handler operable to receive from an external source a first message that includes raw data and that includes a first header having information indicating a destination of the raw data, to extract the raw data from the message, and to load the raw data into the first memory. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed. The message is parsed such that the data is extracted from the message and stored in first memory 312 (Fig.3).

b2) at least one hardwired pipeline operable to process data. See at least the crypto-engines 316 of Fig.3. Note from at least paragraphs [0023], [0044], and [0097], that the system is pipelined.

b3) a pipeline interface operable to retrieve the raw data from the first memory, provide the retrieved raw data to the hardwired pipeline corresponding to the destination, and load processed data from the hardwired pipeline into the second memory. See Fig.3, and note that before the data is processed by units 316, it must be retrieved from the first memory 312. The portion of the system retrieving the data would be the pipeline interface. After processing, the data is loaded into the second memory 318.

b4) an output-data handler operable to retrieve the processed data from the second memory, to generate a second header having first information indicating a destination of the processed data, to generate a second message that includes the processed data and the second header, and to provide the second message to the external source. This is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and

[0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node.

b5) Krishna has not explicitly taught that the pipeline interface is further operable to store a pointer to the processed data in a location associated with destination of the processed data, and wherein the output-data handler is further operable to retrieve the processed data in response to the pointer, and generate the first information from the location. However, this is deemed inherent in Krishna given the output queue of Fig.3. Specifically, data in a queue is accessed via a head pointer. hence, the head pointer must be stored and used to retrieve the data prior to sending it out. Also, the ultimate destination of the data must be stored, otherwise the accelerator will not know where to send the data to. Note also that the first information (outbound destination) must be the same as the second destination (incoming destination) so that the packet ultimately reaches the desired destination.

b6) Krishna has not explicitly taught that the at least one hardwired pipeline is operable to process data without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been

taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

33. Referring to claim 83, Krishna in view of Yin has taught a method as described in claim 43 further comprising:

a) storing in association with the processed data second information indicating the destination of the processed data. This is deemed inherent as when a first processor sends data over a network to a second processor, the destination (second information) is included in the header. Therefore, when this destination is received by the accelerator, it must be stored while the encryption occurs

so that when the data is encrypted, the data is again sent on towards its destination with the destination information.

b) wherein generating the header comprises generating the header in response to the second information. Essentially, after the encryption occurs, the data will be sent on towards its destination. Hence, as is known, a message header including the destination (i.e., first information, which is equal to the second information) will be attached to the data.

34. Referring to claim 84, the method of claim 84 is performed by the accelerator of claim 71. Consequently, claim 84 is rejected for the same reasons set forth in the rejection of claim 71.

35. Referring to claim 85, the method of claim 85 is performed by the accelerator of claim 72. Consequently, claim 85 is rejected for the same reasons set forth in the rejection of claim 72.

36. Claims 6 and 8 are rejected under 35 U.S.C. 103(a) as being unpatentable over Krishna in view of Yin, and further in view of Awoseyi et al., U.S. Patent Application Publication No. US 2003/0231649 A1 (herein referred to as Awoseyi). Furthermore, Wu has been cited as extrinsic evidence showing that fast hardware implementations of cryptography functions are known in the art.

37. Referring to claim 6, Krishna has taught a computing machine, comprising:

a) a processor operable to broadcast a message that includes data and that includes a header having information indicating a destination of the data. See paragraphs [0004], [0010], [0030], [0038], and [0049], along with claims 12-13 of Krishna. The purpose of the accelerator is to receive a message from a first processor on a network, encrypt or decrypt the data, and then send it to another processor over the network.

b) a pipeline accelerator (Fig.2, Fig.3) coupled to the processor and comprising:

b1) a memory. See Fig.3, at least components 312, 302, and 318, all of which may be collectively referred to as "a memory".

b2) a hardwired-pipeline circuit coupled to the memory, including at least one processing pipeline (see Fig.2 and Fig.3, and at least paragraphs [0023], [0044], and [0097]), and operable to:

- receive the message from the processor by receiving the data and the information via at least one same bus line. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed.
- extract the data from the message, load the extracted data into the memory, retrieve the extracted data from the memory, and process the retrieved data with a pipeline corresponding to the destination. See paragraphs [0038] and [0049] an note that the message is parsed such that the data is extracted from the message and stored in buffers 312 (Fig.3). The data is then retrieved from said buffers and processed by crypto-engines 316 (Fig.3).
- Krishna has not explicitly taught that the retrieved data is processed without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for

instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

- Krishna has not taught providing the processed data to the processor. However, Awoseyi has disclosed a situation in which a processor needs to decrypt data prior to performing additional processing, and hence, the encrypted data is sent from

the processor to a decryption engine, at which point the data is decrypted and ultimately sent back to the processor. See paragraph [0032]. Therefore, in order to additionally process data after it has been decrypted, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Krishna such that the processed data is provided to the processor by a decryption engine.

38. Referring to claim 8, Krishna has taught a computing machine comprising:

a) a processor operable to run at least one software application. See Fig.1, component 114, and note that processors inherently run a software application.

b) a pipeline accelerator coupled to the processor (see Fig.1B, component 152, Fig.2, Fig.3, and at least paragraphs [0023], [0044], and [0097]) and comprising:

b1) a memory. See Fig.3, at least components 312, 302, and 318, all of which may be collectively referred to as "a memory".

b2) a hardwired-pipeline circuit coupled to the memory (see Fig.2 and Fig.3, and at least paragraphs [0023], [0044], and [0097]), and operable to:

- receive data from the processor. See Figs.2-3 and note that a data packet is received. According to paragraphs [0004], [0010], and [0030], the accelerator is placed within a network (for instance, in a router) such that it receives data from a processor to perform crypto functions on.
- process the received data, load the processed data into the memory, and retrieve the processed data from the memory. See Fig.3 and note that the data

is processed by crypto-engines 316 before being stored in and ultimately retrieved from output memory 318.

- Krishna has not explicitly taught that the received data is processed without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of

processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

- Krishna has further taught providing the processed data to an external source (see Fig.3), but has not explicitly taught generating a message header that includes, for the processed data, information indicating a destination software application running on the processor, generating a message that includes the retrieved processed data and the message header, and then providing the message to the external source. However, this is deemed inherent given the context of Krishna. Specifically, paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node. Since processors execute software apps, the header indicating a destination processor also inherently indicates the destination software app (i.e., the software app executing on the destination processor).

39. Claims 11 and 46 are rejected under 35 U.S.C. 103(a) as being unpatentable over Krishna in view of Yin and further in view of McLoone et al., U.S. Patent Application Publication No. US 2002/0041685 (herein referred to as McLoone).

40. Referring to claim 11, Krishna in view of Yin has taught a pipeline accelerator as described in claim 9. Krishna has not taught a third memory coupled to the hardwired-pipeline circuit, wherein the hardwired pipeline is operable to generate intermediate data while processing the raw data, and wherein the pipeline interface is operable to load the intermediate data into the third memory and to retrieve the intermediate data from the third memory. However, McLoone has taught a pipelined DES implementation in which intermediate memories are positioned between function rounds. See Fig.3. Such memories are common in pipelined implementations, and McLoone's specific implementation of DES is 9 times faster than non-pipelined, existing techniques. Hence, in order to speed up DES processing, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Krishna to include a third memory coupled to the hardwired-pipeline circuit, wherein the hardwired pipeline is operable to generate intermediate data while processing the raw data, and wherein the pipeline interface is operable to load the intermediate data into the third memory and to retrieve the intermediate data from the third memory.

41. Referring to claim 46, Krishna in view of Yin has taught a method as described in claim 44. Furthermore, the method of claim 46 is performed by the circuit of claim 11. Consequently, claim 46 is rejected for the same reasons set forth in the rejection of claim 11.

42. Claims 13, 47, 68, 75, and 82 are rejected under 35 U.S.C. 103(a) as being unpatentable over Krishna in view of Yin and further in view of Frey et al., U.S. Patent No. 5,185,871 (herein referred to as Frey). Furthermore, Wu has been cited as extrinsic evidence showing that fast hardware implementations of cryptography functions are known in the art.

43. Referring to claim 13, Krishna in view of Kim has taught a pipeline accelerator as described in claim 9. Krishna has not taught an input-data queue coupled to the input-data handler and the pipeline interface, wherein the input-data handler is operable to load into the input-data queue a pointer to a location of the raw data within the first memory and wherein the pipeline interface is operable to retrieve the raw data from the location using the pointer. However, Frey has taught such a concept. See Fig.3, components 17 and 21, and column 11, line 67, to column 12, line 5. Essentially, addresses of operands in a multi-operand buffer are stored in an operand fetch queue so that when it is time to fetch the operands, they are located appropriately. A person of ordinary skill in the art would have recognized that by implementing a multi-packet buffer in Krishna (for buffers 312), multiple packets would be buffered at a time, which ensures that the accelerator always has enough work to do. With such a system, saving the pointer of a packet allows for locating the packet in the buffer. Clearly, when the accelerator is to operate on a packet, the correct packet should be located, and so the location of the packet must be remembered. Therefore, in order to buffer more packets and ensure more work is available, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Krishna to include a multi-packet input buffer and an input-data queue for holding a pointer to a packet (raw data) within the buffer and then using the pointer to locate the desired packet.

44. Referring to claim 47, Krishna in view of Yin has taught a method as described in claim 44. Furthermore, the method of claim 47 is performed by the circuit of claim 13. Consequently, claim 47 is rejected for the same reasons set forth in the rejection of claim 13.

45. Referring to claim 68, Krishna has taught a pipeline accelerator, comprising:

a) a memory. See Fig.3, at least components 312, 302, and 318, all of which may be collectively referred to as "a memory".

b) a hardwired-pipeline circuit coupled to the memory, including at least one processing pipeline (see Fig.2 and Fig.3, and at least paragraphs [0023], [0044], and [0097]), and operable to:

b1) receive a message that includes data and that includes a header having information indicating a destination of the data. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed.

b2) extract the data from the message, load the extracted data into the memory, retrieve the extracted data from the memory, and process the retrieved data with a pipeline corresponding to the destination. See paragraphs [0038] and [0049] an note that the message is parsed such that the data is extracted from the message and stored in buffers 312 (Fig.3). The data is then retrieved from said buffers and processed by crypto-engines 316 (Fig.3).

b3) provide the processed data to an external source. See Fig.3, and note that after processing, the processed data is sent to an external source by way of output FIFO.

b4) extract from the header the information indicating the destination of the data. See paragraph [0038] and claims 12-13 of Krishna.

b5) Krishna has not explicitly taught that the retrieved data is processed without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

b6) Krishna has not taught storing a pointer to the extracted data in a location associated with the pipeline corresponding to the destination, and providing the retrieved data to the pipeline in response to the stored pointer. However, Frey has taught the concept of a multi-value buffer. See Fig.3, components 17 and 21, and column 11, line 67, to column 12, line 5. Essentially, addresses of operands in a multi-operand buffer are stored in an operand fetch queue so that when it is time to fetch the operands, they are located appropriately. A person of ordinary skill in the art would have recognized that by implementing a multi-packet buffer in Krishna (for buffers 312), multiple packets would be buffered at a time, which ensures that the accelerator always has enough work to do. With such a system, saving the pointer of a packet allows for locating the packet in the buffer. Clearly, when the accelerator is to operate on a packet, the correct packet should be located, and so the location of the packet must be remembered. Therefore, in order to buffer more packets and ensure more work is available, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Krishna to include a multi-packet input buffer and an input-data queue to store a pointer to the extracted data in a location associated with the pipeline corresponding to the destination, and provide the retrieved data to the pipeline in response to the stored pointer.

46. Referring to claim 75, Krishna has taught a pipeline accelerator, comprising:

a) first and second memories. See 3, and note first memory 312 and second memory 318.

b) a hardwired-pipeline circuit (see Fig.3, at least components 316) coupled to the first and second memories and comprising:

b1) an input-data handler operable to receive from an external source a first message that includes raw data and that includes a first header having information indicating a destination of the raw data, to extract the raw data from the message, and to load the raw data into the first memory. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed. The message is parsed such that the data is extracted from the message and stored in first memory 312 (Fig.3).

b2) at least one hardwired pipeline operable to process data. See at least the crypto-engines 316 of Fig.3. Note from at least paragraphs [0023], [0044], and [0097], that the system is pipelined.

b3) a pipeline interface operable to retrieve the raw data from the first memory, provide the retrieved raw data to the hardwired pipeline corresponding to the destination, and load processed data from the hardwired pipeline into the second memory. See Fig.3, and note that before the data is processed by units 316, it must be retrieved from the first memory 312. The portion of the system retrieving the data would be the pipeline interface. After processing, the data is loaded into the second memory 318.

b4) an output-data handler operable to retrieve the processed data from the second memory, to generate a second header having first information indicating a destination of the processed data, to generate a second message that includes the processed data and the second header, and to provide the second message to the external source via at least one same bus line. This is deemed inherent given the context of Krishna. Specifically,

paragraphs [0004], [0010], and [0030], set forth that the crypto-chip is located in between nodes on a network (e.g. with a router), and therefore, when sending data to a node over a network, a message header with destination information must be generated such that the data is sent to the appropriate node.

b5) wherein the input data handler is further operable to extract from the header the information indicating the destination of the data. See paragraph [0038] and claims 12-13 of Krishna.

b6) Krishna has not explicitly taught that the at least one hardwired pipeline is operable to process data without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that

taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

b7) Krishna has not taught storing a pointer to the extracted data in a location associated with the pipeline corresponding to the destination, wherein the pipeline interface is further operable to provide the retrieved data to the pipeline in response to the stored pointer. However, Frey has taught the concept of a multi-value buffer. See Fig.3, components 17 and 21, and column 11, line 67, to column 12, line 5. Essentially, addresses of operands in a multi-operand buffer are stored in an operand fetch queue so that when it is time to fetch the operands, they are located appropriately. A person of ordinary skill in the art would have recognized that by implementing a multi-packet buffer in Krishna (for buffers 312), multiple packets would be buffered at a time, which ensures that the accelerator always has enough work to do. With such a system, saving the pointer of a packet allows for locating the packet in the buffer. Clearly, when the accelerator is to operate on a packet, the correct packet should be located, and so the location of the packet must be remembered. Therefore, in order to buffer more packets and ensure more work is available, it would have been obvious to one of ordinary skill in

the art at the time of the invention to modify Krishna to include a multi-packet input buffer and an input-data queue to store a pointer to the extracted data in a location associated with the pipeline corresponding to the destination, and provide the retrieved data to the pipeline in response to the stored pointer.

47. Referring to claim 82, Krishna has taught a method, comprising:

a) receiving a message that includes data and that includes a header having information indicating a destination of the data. See Figs.2-3 and note that a message is received by the input FIFO. This message, according to paragraphs [0038] and [0049] and claims 12-13 of Krishna, includes a header having destination information and data to be processed.

b) extracting the data from the message, loading the extracted data into the memory, retrieving the extracted data from the memory, and processing the retrieved data with a hard-wired pipeline circuit that corresponds to the destination of the data. See paragraphs [0038] and [0049] an note that the message is parsed such that the data is extracted from the message and stored in buffers 312 (Fig.3). The data is then retrieved from said buffers and processed by crypto-engines 316 (Fig.3).

c) providing the processed data to an external source. See Fig.3, and note that after processing, the processed data is sent to an external source by way of output FIFO.

d) extracting from the header the information indicating the destination of the data. See paragraph [0038] and claims 12-13 of Krishna.

e) Krishna has not explicitly taught that the retrieved data is processed without executing a program instruction. However, it should be noted that the purpose of Krishna's acceleration chip is to perform encryption of data prior to sending it out over a network (see the abstract and

paragraph [0010]). It should be further noted that at least the DES or 3DES ciphers are used to encrypt data. See claim 20, for instance. The examiner asserts that hardware implementations of crypto-functions such as DES are well known and advantageous in the art. One such system has been taught by Yin (column 11, lines 1-9), in which an FPGA is programmed to perform DES processing because it is cost-efficient. As is known, if an operation is implemented in FPGA hardware, then instructions are not executed to perform the operation. Instead, the operation is built into the hardware, which is simply reactive to input data. Also known is that hardware implementations are faster than their software counterparts, although they are more limited in flexibility (see Wu, page 1, column 2, last paragraph, which states that it is known that cryptography can be implemented directly in hardware as opposed to with software routines). Herein lies the reason that an FPGA such as that taught by Yin is useful; an FPGA allows for the realization of a fast hardware implementation, but also for flexibility by way of reprogramming the hardware to perform different operations. Hence, since multiple types of processing may be performed by Krishna (see claims 20-21), it would have been obvious to one of ordinary skill in the art to modify Krishna such that the accelerator chip is implemented as an FPGA, wherein the FPGA is cost-effective, fast, and flexible. Such a modification would result in a hardware implementation of the processing so that instruction execution is avoided when processing the data.

f) Krishna has not taught storing a pointer to the extracted data in a location associated with the hard-wired pipeline circuit corresponding to the destination, and providing the retrieved data to the hard-wired pipeline circuit in response to the stored pointer. However, Frey has taught the concept of a multi-value buffer. See Fig.3, components 17 and 21, and column 11, line 67, to

column 12, line 5. Essentially, addresses of operands in a multi-operand buffer are stored in an operand fetch queue so that when it is time to fetch the operands, they are located appropriately. A person of ordinary skill in the art would have recognized that by implementing a multi-packet buffer in Krishna (for buffers 312), multiple packets would be buffered at a time, which ensures that the accelerator always has enough work to do. With such a system, saving the pointer of a packet allows for locating the packet in the buffer. Clearly, when the accelerator is to operate on a packet, the correct packet should be located, and so the location of the packet must be remembered. Therefore, in order to buffer more packets and ensure more work is available, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Krishna to include a multi-packet input buffer and an input-data queue to store a pointer to the extracted data in a location associated with the pipeline corresponding to the destination, and provide the retrieved data to the pipeline in response to the stored pointer.

Allowable Subject Matter

48. Claims 74 and 81 are allowed.

49. Claims 14, 48, 66-67, 73, and 80 objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Conclusion

50. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Applicant is reminded that in amending in response to a rejection of claims, the

patentable novelty must be clearly shown in view of the state of the art disclosed by the references cited and the objections made. Applicant must also show how the amendments avoid such references and objections. See 37 CFR § 1.111(c).

"Chips: New Accelerator Chip From Hi/fn to Speed-Up Virtual Private Networks - "VPNs" - Product Announcement", EDGE Publishing, has taught the Hi/fn 7751 Encryption Processor, a hardware accelerator that encrypts and decrypts data nearly 10 times faster than software implementations.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to DAVID J. HUISMAN whose telephone number is (571)272-4168. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/David J. Huisman/
Primary Examiner, Art Unit 2183
November 6, 2008